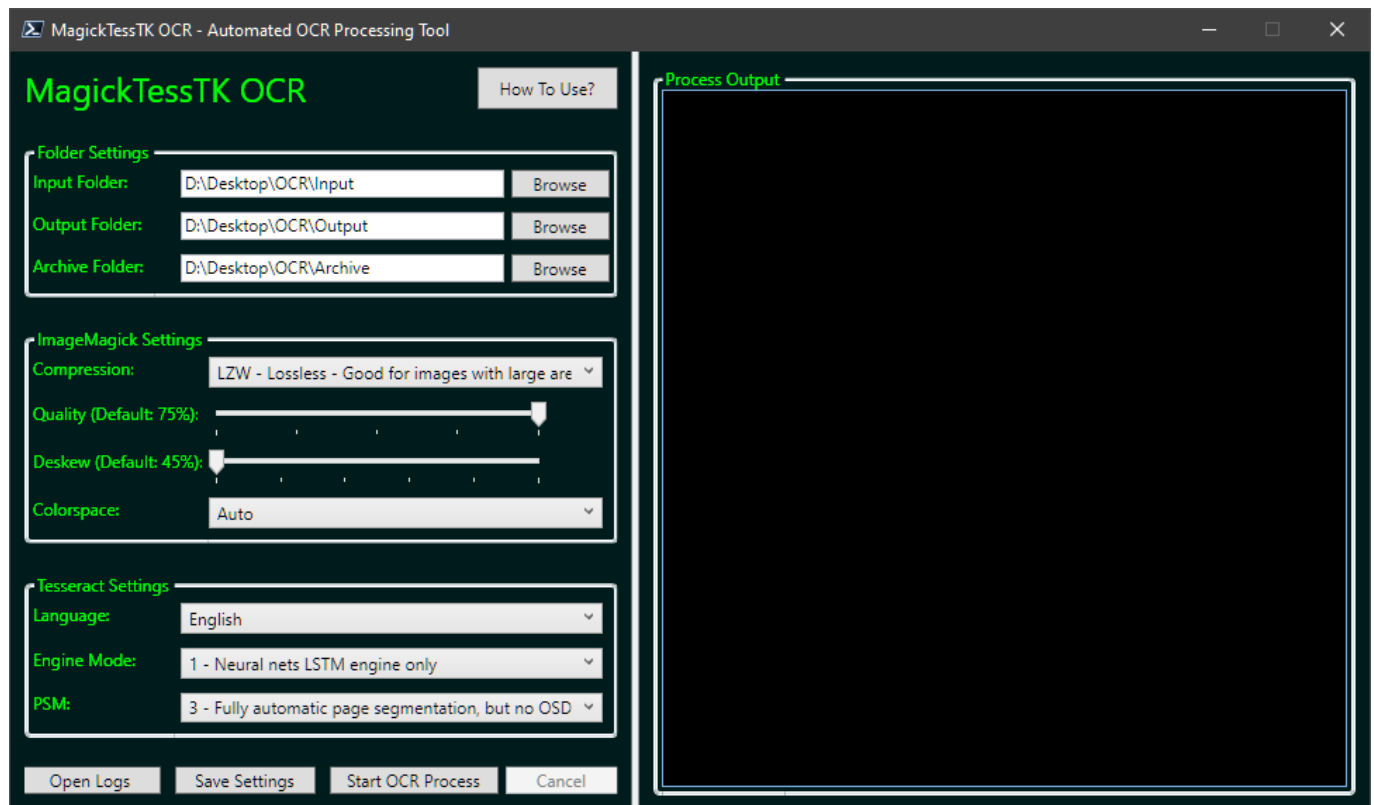


Automated OCR GUI PowerShell 7 Dedicated Script

⚡ **Designed for Batch OCR Process**

🚀 **Batch Folder OCR Only**

This PowerShell script automates OCR processing for batches of different image files, generating searchable PDFs.



Dependencies for MagickTess

The PowerShell script utilizes the following software and models for Tesseract-OCR:

1. **ImageMagick**

- Version: ImageMagick-7.1.1-43-Q16-HDRI-x64-dll.exe

2. **Tesseract-OCR**

- Version: tesseract-ocr-w64-setup-5.5.0.20241111.exe

3. **PDFtk Server**

- Version: pdftk_server-2.02-win-setup.exe

4. **Tessdata for Tesseract-OCR**

- Models: `eng.traineddata`, `enm.traineddata`, `fil.traineddata`

5. PowerShell 7 [For Multithreading Support]

- Version: PowerShell-7.4.6-win-x64.msi

6. GhostScript

- Version: gs10040w64.exe

Folder Structure

When executed, `start_process.bat` will create the following folders:

- **input** – [Batch OCR Only] Place folders containing `.tif` files here. Each folder name becomes the resulting PDF name.
- **archive** – Processed folders from `input` are moved here after OCR.
- **output** – OCR-processed PDF files are saved here.
- **logs** – All process logs are stored here.

Setup & Installation Instructions

1. Download and extract MagickTessTK zip file contents
 - [MagickTessTK v2.0.0.0 Release](#)
2. Navigate to the `setup` folder and run "setup.bat".
3. Run `launcher.bat` to launch the GUI and setup the OCR process.

Folder Structure & PDF Naming

✗ ✗ ✗ Avoid This Structure:

```
input
├── folder1
│   ├── image1.tif ★
│   ├── image2.tif ★
│   ├── subfolder1 <!>
│   │   ├── image1.tif
│   │   └── image2.tif
│   ├── subfolder2
│   │   ├── image1.tif
│   │   └── image2.tif
```

- **Issue:** Files at the root of `folder1` (★) will interrupt processing of subfolders (<!>).
- **Solution:** Ensure `.tif` files are inside subfolders.

✓ ✓ ✓ Proper Folder Structure:

```
input
├── folder1
│   ├── subfolder1 ★
│   │   ├── image1.tif
│   │   └── image2.tif
│   ├── subfolder2 ★
│   │   ├── image1.tif
│   │   └── image2.tif
└── folder2 ★
    ├── image1.tif
    └── image2.tif
```

- **PDF Name:** The folder marked with (★) becomes the PDF name.
- **Example:** `subfolder1` generates `subfolder1.pdf`.

▶ Usage Instructions

1. Place folders containing image files into the `input` directory.

Supported Image Extensions/Types:

```
.bmp  .jpeg  .gif  .png
.dib  .jpe   .tif  .heic
.jpg  .jiff   .tiff
```

2. Run `start_process.bat` and wait for the process to complete.
3. OCR-processed PDF files will be saved in the `output` directory.

📄 PDF Naming Convention







- `folder1/subfolder1/many tif files here` → `subfolder1.pdf`
- `folder1/subfolder2/many tif files here` → `subfolder2.pdf`
- `folder2/many tif files here` → `folder2.pdf`

🤖 MagickTessTK Features List:





Core Features

- 🗄️ Single instance execution with mutex handling
- 🏗️ Multi-threaded parallel processing
- 📁 Structured folder organization (input/output/archive)
- 📝 Comprehensive logging with color-coded output
- ⚙️ Configurable settings via INI file




Image Processing (ImageMagick)

-  Support for multiple image formats (BMP, JPEG, PNG, TIFF, HEIC, etc.)
-  Image integrity verification
-  Automatic image deskewing
-  Configurable color space processing
-  Customizable compression settings
-  Quality control parameters





OCR Processing (Tesseract)

-  Multi-language OCR support
-  Configurable OCR engine modes
-  Page segmentation optimization
-  PDF output generation





PDF Processing (PDFtk)

-  Efficient chunk-based PDF merging
-  Automatic cleanup of intermediate files
-  Sequential page ordering





File Management

-  Automatic archive organization
-  Empty folder cleanup
-  Corrupted file detection and handling
-  Subfolder structure preservation

Monitoring & Reporting

-  Processing statistics tracking
-  Performance timing
-  Detailed error logging
-  File integrity reporting

Safety Features

-  Input file preservation
-  Corrupt file isolation
-  Process interruption handling
-  Resource cleanup on exit